**23. Характеристики алгоритма Sapphire**

Jones, Hosking, Moss — The Garbage collection handbook

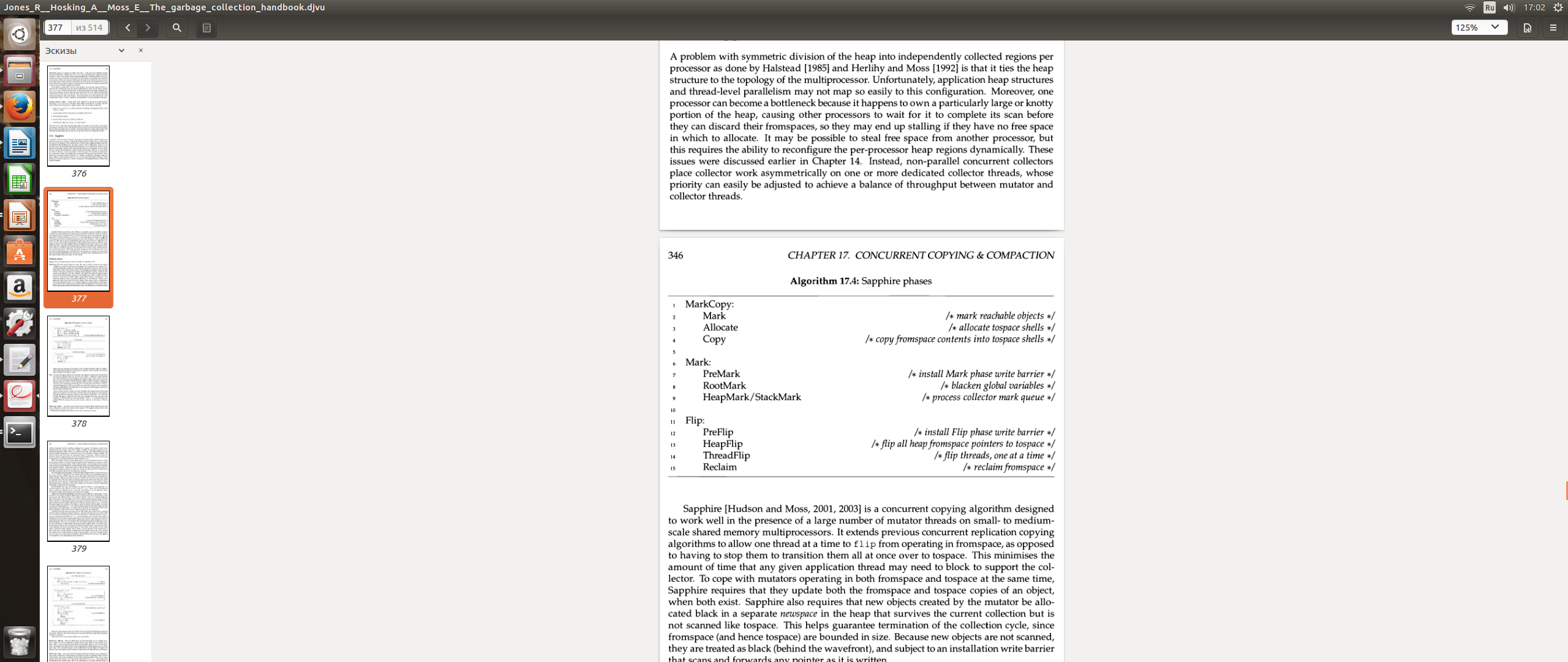
17.6 Sapphire стр. 345

Эквивалентность объектов в “From” и “To”

* Объекты в целевом и исходном пространстве гарантированно имеют одинаковые значения полей
* Реализуется при помощи Write барьера обновляющего значения полей в обоих пространствах при наличии ссылки на новое положение в заголовке объекта

Не-параллельный конкурентный сборщик выполняет работу ассиметрично на одной и более выделенных нитей сборщика, чья очередность может быть легко отрегулирована, дабы достичь баланса между нитями мутатора и сборщика.

Фазы алгоритма:

Sapphire — конкурентный копирующий алгоритм, разработанный чтобы эффективно работать в присутствии большого количества нитей мутатора на small- medium-scale мультипроцессорах с разделяемой памятью. Он позволяет одной нити за раз сделать кувырок(flip), перенеся деятельность из пространства fromspace, в противовес тому, чтобы остановить их для перенесения их всех в пространство tospace. Это минимизирует количество времени, на которое любая нить данного приложения может быть заблокирована, дабы сборщик отработал. Чтобы совладать с мутаторами, действующими одновременно и в tospace и с fromspace одновременно, требуется, чтобы они обновляли обе копии объекта, когда обе существуют. Sapphire также требует, чтобы новые объекты, созданные мутатором размещались бы черными(см. трехцветная раскраска) в отдельном newspace в куче, которое переживает текущую сборку, но не сканируется как tospace. Это помогает гарантировать завершение цикла сборки, поскольку fromspace (и tospace) ограничены в размерах. Поскольку новые объекты не сканируются, к ним относятся как к черным, и ставят write барьер который сканирует и пересылает любой указатель, если он записывается